Component Based Product Line Engineering With Uml Component Based Development Series

Many approaches to creating Software Product Lines have emerged that are based on Model-Driven Engineering. This book introduces both Software Product Lines and Model-Driven Engineering, which have separate success stories in industry, and focuses on the practical combination of them. It describes the challenges and benefits of merging these two software development trends and provides the reader with a novel approach and practical mechanisms to improve software development productivity. The book is aimed at engineers and students who wish to understand and apply software product lines and model-driven engineering in their activities today. The concepts and methods are illustrated with two product line examples: the classic smart-home systems and a collection manager information system.

"Designing Software Product Lines with UML is well-written, informative, and addresses a very important topic. It is a valuable contribution to the literature in this area, and offers practical guidance for software architects and engineers." --Alan Brown Distinguished Engineer, Rational Software, IBM Software Group "Gomaa''s process and UML extensions allow development teams to focus on feature-oriented development and provide a basis for improving the level of reuse across multiple software development efforts. This book will be valuable to any software development professional who needs to manage across projects and wants to focus on creating software that is consistent, reusable, and modular in nature." --Jeffrey S Hammond Group Marketing Manager, Rational Software, IBM Software Group "This book brings together a good range of concepts for understanding software product lines and provides an organized method for developing product lines using object-oriented techniques with the UML. Once again, Hassan has done an excellent job in balancing the needs of both experienced and novice software engineers." --Robert G. Pettit IV, Ph.D. Adjunct Professor of Software Engineering, George Mason University "This breakthrough book provides a comprehensive step-by-step approach on how to develop software product lines, which is of great strategic benefit to industry. The development of software product lines enables significant reuse of software architectures. Practitioners will benefit from the well-defined PLUS process and rich case studies." --Hurley V. Blankenship II Program Manager, Justice and Public Safety,

Science Applications International Corporation "The Product Line UML based Software engineering (PLUS) is leading edge. With the author''s wide experience and deep knowledge, PLUS is well harmonized with architectural and design pattern technologies." --Michael Shin Assistant Professor, Texas Tech University Long a standard practice in traditional manufacturing, the concept of product lines is quickly earning recognition in the software industry. A software product line is a family of systems that shares a common set of core technical assets with preplanned extensions and variations to address the needs of specific customers or market segments. When skillfully implemented, a product line strategy can yield enormous gains in productivity, quality, and time-to-market. Studies indicate that if three or more systems with a degree of common functionality are to be developed, a product-line approach is significantly more cost-effective. To model and design families of systems, the analysis and design concepts for single product systems need to be extended to support product lines. Designing Software Product Lines with UML shows how to employ the latest version of the industry-standard Unified Modeling Language (UML 2.0) to reuse software requirements and architectures rather than starting the development of each new system from scratch. Through real-world case studies, the book illustrates the fundamental concepts and technologies used in the design and implementation of software product lines. This book describes a new UML-based software design method for product lines called PLUS (Product Line UML-based Software engineering). PLUS provides a set of concepts and techniques to extend UML-based design methods and processes for single systems in a new dimension to address software product lines. Using PLUS, the objective is to explicitly model the commonality and variability in a software product line. Hassan Gomaa explores how each of the UML modeling views--use case, static, state machine, and interaction modeling--can be extended to address software product families. He also discusses how software architectural patterns can be used to develop a reusable component-based architecture for a product line and how to express this architecture as a UML platform-independent model that can then be mapped to a platform-specific model. Key topics include: Software product line engineering process, which extends the Unified Development Software Process to address software product lines Use case modeling, including modeling the common and variable functionality of a product line Incorporating feature modeling into UML for modeling common, optional, and alternative product line features Static modeling, including modeling the boundary of the product line and information-intensive entity classes Dynamic

modeling, including using interaction modeling to address use-case variability State machines for modeling state-dependent variability Modeling class variability using inheritance and parameterization Software architectural patterns for product lines Component-based distributed design using the new UML 2.0 capability for modeling components, connectors, ports, and provided and required interfaces Detailed case studies giving a step-by-step solution to real-world product line problems Designing Software Product Lines with UML is an invaluable resource for all designers and developers in this growing field. The information, technology, and case studies presented here show how to harness the promise of software product lines and the practicality of the UML to take software design, quality, and efficiency to the next level. An enhanced online index allows readers to quickly and easily search the entire text for specific topics. Software product line engineering has proven to be the methodology for developing a diversity of software products and software intensive systems at lower costs, in shorter time, and with higher quality. In this book, Pohl and his co-authors present a framework for software product line engineering which they have developed based on their academic as well as industrial experience gained in projects over the last eight years. They do not only detail the technical aspect of the development, but also an integrated view of the business, organisation and process aspects are given. In addition, they explicitly point out the key differences of software product line engineering compared to traditional single software system development, as the need for two distinct development processes for domain and application engineering respectively, or the need to define and manage variability.

A cutting-edge, UML-based approach to software development and maintenance that integrates component-based and product-line engineering methods. - ripe market: development of component-based technologies is a major growth area - CBD viewed as a faster, more flexible way of building systems that can easily be adapted to meet rapidly-changing business needs and integrate legacy and new applications (e.g. Forrester report in June 1998 predicted that by 2001 "half of packaged apps vendors will deliver component-based apps"; e.g. Butler Group Management Briefing (2000): "Butler Group is now advising that all new-build and significant modification activity should be based on component architectures...Butler Group belives that Component-Based Development is one of the most important events in the evolution of information technology" e.g. Gartner Group estimates that "by 2003, 70% of new applications will be deployed as a combination of pre-assembled and newly created components

integrated to form complex business-systems. The book defines, describes and shows how to use a method for component-based product-line engineering, supported by UML. This method aims to dramatically increase the level of reuse in software development by integrating the strengths of both of these approaches. UML is used to describe components during the analysis, design & implementation stages and capture their characteristics and relationships.This method includes two new kinds of extensions to the UML: new stereotypes to capture KobrA-specific concepts and new metamodel elements to capture variabilities. The method makes components the focus of the entire software development process, not just the implementation and deployment phases. The method has grown out of work by two companies in industry (Softlab & Psipenta) and two research organizations (GMD FIRST & Fraunhofer IESE) called the KobrA project. It is influenced by a number of successful existing methods e.g. Fusion method, Cleanroom method, Catalysis & Rational Unified Process, integrated with new ideas in an innovative way. Benefits for the reader: - gain a clear understanding of the product-line and component-based approaches to software development - learn how to use UML to describe components in analysis, design and implementation of components - learn how to develop and apply component-based frameworks in product-lines - learn how to build new systems from pre-existing components and ensure that components are of a high quality The book also includes: - case studies: library system example running throughout the chapters; ERP/business software system as appendix or separate chapter - bibliography - glossary - appendices covering: UML profiles, concise process description in the form of UML activity diagrams, refinement/translation patterns AUDIENCE Software engineers, architects & project managers. Software engineers working in the area of distributed/enterprise systems who want a method for applying a component-based or product-line engineering approach in practice.

9th International Conference on Software Reuse, ICSR 2006, Torino, Italy, June 12-15, 2006, Proceedings

A Family-based Software Development Process

18th European Conference, EuroSPI 2011, Roskilde, Denmark, June 27-29, 2011, Proceedings

QOSPL

On the Move to Meaningful Internet Systems: OTM 2011 Workshops

Systems product line engineering handbook

Integrating three important aspects of software reuse--technical, management, and organizational--this indispensable reference shows how these fundamental aspects are used in the development lifecycle of component-based software engineering and product line engineering. The book explores the basic foundations upon which reuse processes and approaches can be established and discusses state of the art and state of the practice of software reuse.

Model-Driven Software Development (MDSD) is currently a highlyregarded development paradigm among developers and researchers.With the advent of OMG's MDA and Microsoft's Software Factories,the MDSD approach has moved to the centre of the programmer'sattention, becoming the focus of conferences such as OOPSLA, JAOOand OOP. MDSD is about using domain-specific languages to create models thatexpress application structure or behaviour in an efficient anddomain-specific way. These models are subsequently transformed intoexecutable code by a sequence of model transformations. This practical guide for software architects and developers ispeppered with practical examples and extensive case studies.International experts deliver: * A comprehensive overview of MDSD and how it relates to industrystandards such as MDA and Software Factories. * Technical details on meta modeling, DSL construction,model-to-model and model-to-code transformations, and softwarearchitecture. * Invaluable insight into the software development process, plusengineering issues such as versioning, testing and product lineengineering. * Essential management knowledge covering economic andorganizational topics, from a global perspective. Get started and benefit from some practical support along the way!

Component-based Product Line Engineering with UMLAddison-Wesley Professional

A Product Line is a set of products with common elements and variable features. Including Product Lines in an overall development strategy tailored to the commercial and/or industrial context delivers significant benefits: products that are more suitable, reduction in cost, shorter development timescales, quality improvement, etc. This work, Systems Product Line Engineering, brings together a summary of the state-of-the-art with lessons learnt from industrial experience in implementing Product Lines of various kinds, in terms of marketplace, number of applications, degree of variability, etc. It is resolutely practical, and is intended to complement existing Systems Engineering manuals; indeed, it adopts the same process structures. It includes: • Definitions and examples: Product Line, Product Lines organizations, Product Line Engineering, • Processes, from needs analysis through to disposal, • Systems Engineering methods, particularly Model-Based Product Line Systems Engineering, • Organization: development in silos, development in platforms, • Implementation strategies and management processes. This work is intended for practitioners: engineers, project managers, instructors, researchers, students and developments of systems that fit into this approach. Elected Incose Product of the Year 2015.

Practices and Patterns: Practices and Patterns

Experience and Research Directions

Case Studies

Reuse of Off-the-Shelf Components

4th International Workshop, PFE 2001 Bilbao, Spain, October 3-5, 2001 Revised Papers

14th International Conference, SPLC 2010, Jeju Island, South Korea, September 13-17, 2010. Proceedings

This book constitutes the refereed proceedings of the 8th

International Conference on Software Reuse, ICSR-8, held in Madrid, Spain in July 2004. The 28 revised full papers presented were carefully reviewed and selected from numerous submissions. The papers are organized in topical sections on software variability: requirements; testing reusable software; feature modeling; aspect-oriented software development; component and service development; code level reuse; libraries, classification, and retrieval; model-based approaches; transformation and generation; and requirements.
This book contains the proceedings of the 5th International Workshop on Product Family Engineering, PFE-5. This workshop was held in Siena, Italy, November 4–6, 2003. This workshop was the fifth in the series, with the same subject, software product family engineering. These workshops have been held initially irregularly about every 18 months since 1996.
Since 1999 the workshop has been held every second year in the fall. The proceedings of the second, third and fourth workshops were published as Springer LNCS volumes 1429, 1951 and 2290. The workshops were organized within co-operation projects of European industry. The first two were organized by ARES (Esprit IV 20.477) 1995–1999; this project had 3 industrial and 3 academic partners, and studied software architectures for product families. Some of the partners continued in the ITEA project if99005 ESAPS (1999–2001).
ITEA is the software development programme (?! 2023) within the European Eureka initiative. ITEA projects last for 2 years, and ESAPS was succeeded by CAFÉ (ITEA if00004) for 2001–2003 and FAMILIES (ITEA if02009). This fifth workshop was initially prepared within CAFÉ and the preparation continued in FAMILIES. As usual Henk Obbink was the workshop chair, and Linda Northrop and Sergio Bandinelli were the co-chairs.
Over the last decade, software product line engineering (SPLE) has emerged as one of the most promising software development paradigms for increasing productivity in IT-related industries. Detailing the various aspects of SPLE implementation in different domains, Applied Software Product Line Engineering documents best practices with regard to system development. Expert contributors from academia and industry come together and focus on core asset development, product development, and management, addressing the process, technical, and organizational issues needed to meet the growing demand for information. They detail the

adoption and diffusion of SPLE as a primary software development paradigm and also address technical and managerial issues in software product line engineering. Providing an authoritative perspective of the latest research and practice in SLPE, the text: Presents in-depth discussions and many industry / case studies Covers applications in various domains including automotive, business process management, and defense Organized according to the organizational, process, and technical aspects of software product lines within an organization Provides the expertise of a distinguished panel of global contributors Ever-increasing global competition coupled with a fragile world economy means that the pressure is on for software engineers and software process improvement professionals to find ways to meet the needs of expanding markets—with greater efficiency and effectiveness. This book arms readers with the insight needed to harness the power of SPLE to increase productivity, reduce time to market, and to handle the growing diversity in the quickly evolving global marketplace.

This book constitutes the thoroughly refereed post-proceedings of the international conference NetObjectDays 2002, held in Erfurt, Germany, in October 2002. The 26 revised full papers presented were carefully selected during two rounds of reviewing and revision. The papers are organized in topical sections on embedded and distributed systems; components and MDA; Java technology; Web services; aspect-oriented software design; agents and mobility; software product lines; synchronization; testing, refactoring, and CASE tools.

7th International Conference, ICSR-7, Austin, TX, USA, April 15-19, 2002. Proceedings

Component-Based Software Testing with UML

The Development of Component-based Information Systems

7th International Symposium, CBSE 2004, Edinburgh, UK, May 24-25, 2004, Proceedings

Objects, Components, Architectures, Services, and Applications for a Networked World

Software Product Lines

In the past two years, the Smalltalk and Java in Industry and Education C- ference (STJA) featured a special track on generative programming, which was organized by the working group \Generative and Component-Based Software Engineering" of the \Gesellschaft fur ¨

Informatik" FG 2.1.9 \Object-Oriented Software Engineering." This track covered a wide range of related topics from domain analysis, software system family engineering, and software product - nes, to extendible compilers and active libraries. The talks and keynotes directed towards this new software engineering paradigm received much attention and - terest from the STJA audience. Hence the STJA organizers suggested enlarging this track, making it more visible and open to wider, international participation. This is how the GCSE symposium was born. The rst GCSE symposium attracted 39 submissions from all over the world. This impressive number demonstrates the international interest in generative programming and related elds. After a careful review by the program comm- tee, fteen papers were selected for presentation. We are very grateful to the members of the program committee, all of them renowned experts, for their dedication in preparing thorough reviews of the submissions. Special thanks go to Elke Pulvermuller ¨ and Andreas Speck, who proposed and organized a special conference event, the Young Researches Workshop (YRW). This workshop provided a unique opportunity for young scientists and Ph.D.

This book constitutes the refereed proceedings of the 5th European Conference on Software Architecture, ECSA 2011, held in Essen, Germany, in September 2011. The 13 revised full papers presented together with 24 emerging research papers, and 7 research challenge poster papers were carefully reviewed and selected from over 100 submissions. The papers are organized in topical sections on requirements and software architectures; software architecture, components, and compositions; quality attributes and software architectures; software product line architectures; architectural models, patterns and styles; short papers; process and management of architectural decisions; software architecture run-time aspects; ADLs and metamodels; and services and software architectures.

Component-Based Software Engineering (CBSE) is the way to produce software fast. This book presents the concepts in CBSE. While detailing both the advantages and the limitations of CBSE, it covers every aspect of component engineering, from software engineering practices to the design of software component infrastructure, technologies, and system.

This volume constitutes the refereed proceedings of the 18th EuroSPI conference, held in Roskilde, Denmark, in June 2011. The 18 revised full papers presented together with 9 key notes were carefully reviewed and selected. They are organized in topical sections on SPI and assessments; SPI and implementation; SPI and improvement methods; SPI organization; SPI people/ teams; SPI and reuse; selected key notes for SPI implementation.

Reuse Based Software Engineering

Methods and Techniques

8th International Conference, ICSR 2004, Madrid, Spain, July 5-9, 2004, Proceedings

Confederated International Workshops and Posters, EI2N+NSF ICE, ICSP+INBAST, ISDE, ORM, OTMA, SWWS+MONET+SeDeS, and VADER 2011, Hersonissos, Crete, Greece, October 17-21, 2011, Proceedings

Software Product Lines: Going Beyond
Component-based Software Engineering

Annotation This work provides a comprehensive overview of research and practical issues relating to component-based information systems (CBIS). Spanning the organizational, developmental, and technical aspects of the subject, the original research included here provides fresh insights into successful CBIS technology and application, including the selection and trading of commercial off-the shelf products (COTS).

This book constitutes the thoroughly refereed post-proceedings of the 4th International Workshop on Product Family Engineering, PFE 2001, held in Bilbao, Spain, in October 2001. The 31 revised full papers presented together with an introduction and six session reports were carefully reviewed and selected for inclusion in the book. The papers are organized in topical sections on product issues, process issues, community issues, platform and quality solutions, diversity solutions, product validation, and process validation.

Software product lines are emerging as an important new paradigm for so- ware development. Product lines are enabling organizations to achieve impressive time-to-market gains and cost reductions. In 1997, we at the Software Engine- ing Institute (SEI) launched a Product Line Practice Initiative. Our vision was that product line development would be a low-risk, high-return proposition for the entire software engineering community. It was our hope from the beginning that there would eventually be su?cient interest to hold a conference. The First Software Product Line Conference (SPLC1) was the realization of that hope. Since SPLC1, we have seen a growing interest in software product lines. Companies are launching their own software product line initiatives, product line technical and business practices are maturing, product line tool vendors are emerging, and books on product lines are being published. Motivated by the enthusiastic response to SPLC1 and the increasing number of software product lines and product line researchers and practitioners, the SEI is proud to sponsor this second conference dedicated to software product lines. We were grati?ed by the submissions to SPLC2 from all parts of the globe, from government and commercial organizations. From these submissions we were able to assemble a rich and varied conference program with unique opportunities for software product line novices, experts, and those in between. This collection represents the papers selected from that response and includes research and experience reports.

The book describes a method for developing the testing of components in parallel with their functionality based on models. UML models are used to derive the testing architecture for an application, the testing interfaces and the component testers. The method provides a process and guidelines for modeling and developing these artifacts. The book also discusses the implications of built-in contract testing with other component-based development technologies such as product-line engineering, middleware platforms, reuse principles etc. Still further, it describes a new method for specifying and checking real-time properties of object-oriented, component-based real-time systems that are based on dynamic execution time analysis with optimization algorithms.

5th International Workshop, PFE 2003, Siena, Italy, November 4-6, 2003, Revised Papers

Techniques, Organizations, and Measurement

Model-Driven Software Development

Software Architecture

Development of Component-based Information Systems

Software Product-Family Engineering

Industrial development of software systems needs to be guided by recognized engineering principles. Commercial-off-the-shelf (COTS) components enable the systematic and cost-effective reuse of prefabricated tested parts, a characteristic approach of mature engineering disciplines. This reuse necessitates a thorough test of these components to make sure that each works as specified in a real context. Beydeda and Gruhn invited leading researchers in the area of component testing to contribute to this monograph, which covers all related aspects from testing components in a context-independent manner through testing components in the context of a specific system to testing complete systems built from different components. The authors take the viewpoints of both component developers and component users, and their contributions encompass functional requirements such as correctness and functionality compliance as well as non-functional requirements like performance and robustness. Overall this monograph offers researchers, graduate students and advanced professionals a unique and comprehensive overview of the state of the art in testing COTS components and COTS-based systems.

As a result of the open-source movement there is now a great deal of reusable software available in the public domain. This offers significant functionality that commercial software vendors can use in their software projects. Open-source approaches to software development have illustrated that complex, mission critical software can be developed by distributed teams of developers sharing a common goal. Commercial software vendors have an opportunity to both learn from the op- source community as well as leverage that knowledge for the benefit of its commercial clients. Nonetheless, the open-source movement is a diverse collection of ideas, knowledge, techniques, and solutions. As a result, it is far from clear how these approaches should be applied to commercial software engineering. This paper has looked at many of the dimensions of the open-source movement, and provided an analysis of the different opportunities available to commercial software vendors. References and Notes 1. It can be argued that the open-source community has produced really only two essential 9 products -- Apache (undeniably the most popular web server) and Linux although both are essentially reincarnations of prior systems. Both are also somewhat products of their times: Apache filled a hole in the then emerging Web, at a time no platform vendor really knew how to step in, and Linux filled a hole in the fragmented Unix market, colored by the community s general anger against Microsoft. 2.Evans Marketing Services, Linux Developers Survey, Volume 1, March 2000.

This work provides a comprehensive overview of research and practical issues relating to component-based development information systems (CBIS). Spanning the organizational, developmental, and technical aspects of the subject, the original research included here provides fresh insights into successful CBIS technology and application. Part I covers component-based development methodologies and system architectures. Part II analyzes different aspects of managing component-based development. Part III investigates component-based development versus commercial off-the-shelf products (COTS), including the selection and trading of COTS products.

This book constitutes the refereed proceedings of the 7th International Symposium on Component-Based Software Engineering, CBSE 2004, held in Edinburgh, UK in May 2004 as an adjunct event to ICSE 2004. The 12 revised long papers and 13 revised short papers presented together with the abstracts of 2 invited talks were carefully reviewed and selected from 82 submissions. The papers are organized in topical sections on generation and adoptation of component-based systems, tools and building frameworks, components for real-time embedded systems, extra-functional properties of components and component-based systems, and measurement and prediction models for component assemblies.

Testing Commercial-off-the-Shelf Components and Systems

Putting the Pieces Together

Component-Based Software Quality

Applied Software Product Line Engineering

Software Reuse: Methods, Techniques, and Tools

Component-based Software Development

- First book of its kind (case studies in CBD) - Covers different kinds of components - Covers different component models/technologies - Includes a wide scope of CBD topics - Covers both theoretical and practical work - Includes both formal and informal approaches - Provides a snapshot of current concerns and pointers to future trends

This book covers research into the most important practices in product line organization. Contributors offer experience-based knowledge on the domain and application engineering, the modeling and management of variability, and the design and use of tools to support the management of product line-related knowledge.

Component-based software development, CBSD, is no longer just one more new paradigm in software engineering, but is effectively used in development and practice. So far, however, most of the efforts from the software engineering community have concentrated on the functional aspects of CBSD, leaving aside the treatment of the quality issues and extra-functional properties of software components and component-based systems. The 16 revised chapters presented were carefully reviewed and

selected for inclusion in the book; together with an introductory survey, they give a coherent and competent survey of the state of the art in the area. The book - the first to focus on quality issues of components and component-based systems - is organized in topical parts on COTS selection, testing and certification, software component quality models, formal models to quality assessment, and CBSD management.

During the last few years, software evolution research has explored new domains such as the study of socio-technical aspects and collaboration between different individuals contributing to a software system, the use of search-based techniques and meta-heuristics, the mining of unstructured software repositories, the evolution of software requirements, and the dynamic adaptation of software systems at runtime. Also more and more attention is being paid to the evolution of collections of inter-related and inter-dependent software projects, be it in the form of web systems, software product families, software ecosystems or systems of systems. With this book, the editors present insightful contributions on these and other domains currently being intensively explored, written by renowned researchers in the respective fields of software evolution. Each chapter presents the state of the art in a particular topic, as well as the current research, available tool support and remaining challenges. The book is complemented by a glossary of important terms used in the community, a reference list of nearly 1,000 papers and books and tips on additional resources that may be useful to the reader (reference books, journals, standards and major scientific events in the domain of software evolution and datasets). This book is intended for all those interested in software engineering, and more particularly, software maintenance and evolution. Researchers and software practitioners alike will find in the contributed chapters an overview of the most recent findings, covering a broad spectrum of software evolution topics. In addition, it can also serve as the basis of graduate or postgraduate courses on e.g., software evolution, requirements engineering, model-driven software development or social informatics.

International Conference NetObjectDays, NODe 2002, Erfurt, Germany, October 7-10, 2002, Revised Papers

Systems, Software and Services Process Improvement

Model-Driven and Software Product Line Engineering

Generative and Component-Based Software Engineering

Research Issues in Engineering and Management

This volume constitutes the refereed proceedings of the 14th International Software Product Line Conference, SPLC 2010, held on Jeju Island, South Korea, in September 2010.

Component-based software development (CBD) is an emerging discipline that promises to take software engineering into a new era. Building on the achievements of object-oriented software construction, CBD aims to deliver software engineering from a cottage industry into an industrial age for Information Technology, wherein software can be assembled from components, in the manner that hardware systems are currently constructed from kits of parts. This volume provides a survey of the current state of CBD, as reflected by activities that have been taking place recently under the banner of CBD, with a view to giving pointers to future trends. The contributions report case studies - self-contained, fixed-term investigations with a finite set of clearly defined objectives and measurable outcomes - on a sample of the myriad aspects of CBD. The book includes chapters dealing with COTS (commercial off-the-shelf) components; methodologies for CBD; compositionality, i.e. how to calculate or predict properties of a composite from those of its constituents; component software testing; and grid computing.

In his study, Mahdi Derakhshanmanesh builds on the state of the art in modeling by proposing to integrate models into running software on the component-level without translating them to code. Such so-called model-integrating software exploits all advantages of models: models implicitly support a good separation of concerns, they are self-documenting and thus improve understandability and maintainability, and in contrast to model-driven approaches there is no synchronization problem anymore between the models and the code generated from them. Using model-integrating components, software will be easier to build and easier to evolve by just modifying the respective model in an editor. Furthermore, software may also adapt itself at runtime by transforming its own model part. Contents An extensive description of foundations and related work A realization concept for Model-Integrating Software Components (MoCos) A reference implementation (based on Java, OSGi and TGraphs) and its application Target Groups Software engineering researchers and students in the fields of component-based software engineering, self-adaptive software, (dynamic) software product line engineering and modelling Component developers and users as well as creators of domain-specific (modeling) languages About the Author Mahdi Derakhshanmanesh is a postdoctoral researcher and lecturer at the University

of Koblenz-Landau, Institute for Software Technology.
This book constitutes the refereed proceedings of the 9th International Conference on Software Reuse, ICSR 2006, held in Torino, Italy, in June 2006. The book presents 27 revised full papers and 13 revised short papers, carefully reviewed and selected from numerous submissions. The Coverage includes COTS selection and integration; product lines, domain analysis, and variability; reengineering maintenance; programming languages and retrieval; aspect-oriented software development; approaches and models; and components.
Foundations, Principles and Techniques
Evolving Software Systems
Technology, Engineering, Management
5th European Conference, ECSA 2011, Essen, Germany, September 13-16, 2011. Proceedings
Software Product Line Engineering
12th International Symposium, CBSE 2009 East Stroudsburg, PA, USA, June 24-26, 2009 Proceedings

The European Conference on Software Architecture (ECSA) is the premier European conference dedicated to the field of software architecture, covering all architectural features of software engineering. It is the follow-up of a successful series of European workshops on software architecture held in the UK in 2004 (Springer LNCS 3047), Italy in 2005 (Springer LNCS 3527), and France in 2006 (Springer LNCS 4344). It evolved into a series of European conferences whose first edition was ECSA 2007, held in Madrid, Spain during September 24–26, 2007 (Springer LNCS 4758). This year's conference was held at the beautiful Coral Beach Hotel and Resort near Paphos in Cyprus. As with the previous versions of the conference, ECSA 2008 (Springer LNCS 5292) provided an international forum for researchers and practitioners from academia and industry to present innovative research and to discuss a wide range of topics in the area of software architecture. It focused on formalisms, technologies, and processes for describing, verifying, validating, transforming, building, and evolving software systems. Covered topics included architecture modelling, architecture description languages, architectural aspects, architecture analysis, transformation and synthesis, architecture evolution, quality attributes, model-driven engineering, built-in testing and architecture-based support for component-based and service-oriented systems. The conference attracted paper submissions from 29 countries (Australia, Belgium, Brazil, Canada, China, Chile, Denmark, Finland, France, Germany, Greece, Ireland, Italy, Lithuania, Luxembourg, Malta, The Netherlands, Norway, Pakistan, Peru, - land, Portugal, Romania, South Africa, Spain, Turkey, the UK, USA, and Venezuela).

The 2009 Symposium on Component-Based Software Engineering (CBSE 2009) was the 12thin a series ofsuccessful eventsthat havegrowninto the main forum for industrial and academic experts to discuss component technology. Component-based software engineering (CBSE) has emerged as the under- ing technology for the assembly of ?exible software systems. In essence, CBSE is about composing computational building blocks to construct larger building blocks that ful?ll client needs. Most software engineers are involved in some form of component-based development. Nonetheless, the implications of CBSE adoption are wide-reaching and its challenges

grow in tandem with its uptake, continuing to inspire our scienti?c speculation. Component-based development necessarily involves elements of software - chitecture, modular software design, software veri?cation, testing, con?guration and deployment. This year's submissions represent a cross-section of CBSE - search that touches upon all these aspects. The theoretical foundations of c- ponent speci?cation, composition, analysis, and veri?cation continue to pose research challenges. What exactly constitutes an adequate semantics for c- munication and composition so that bigger things can be built from smaller things? How can formal approaches facilitate predictable assembly through b- ter analysis? We have grouped the proceedings into two sub-themes that deal with these issues: component models and communication and composition. At the same time, the world is changing.

This volume constitutes the refereed proceedings of nine international workshops, EI2N+NSF ICE, ICSP, INBAST, ISDE, MONET, ORM, SeDeS, SWWS, and VADER 2011, held as part of OTM 2011 in Hersonissos on the island of Crete, Greece, in October 2011. The 64 revised full papers presented were carefully reviewed and selected from a total of 104 submissions. The volume also includes three papers from the On the Move Academy (OTMA) 2011 and five ODBASE 2011 poster papers. Topics of the workshop papers are enterprise integration and semantics, information centric engineering, interoperability, industrial and business applications of semantic Web applications, information systems in distributed environments, process management in distributed information system development, distributed information systems: implementation issues and applications, industrial applications of fact-oriented modeling, data warehouse modeling, extensions to fact-oriented modeling, model validation procedures, schema transformations and mapping, semantic Web and Web semantics, ontology development, deployment and interoperability, data access and efficient computation, efficient information processing, exchange and knowledge synthesis algorithms, mobile and networking technologies for social applications, semantic and decision support, variability in software architecture, and dynamic and adaptive architectures.

The authors outline a systematic method for rapid software production through the family-oriented abstraction, specification, and translation (FAST) process. FAST uses practical domain engineering to decrease the time and effort necessary to develop, deliver, and maintain software. Any software development projects using C, C++, or Java can incorporate the FAST model. The CD-ROM contains a FAST PASTA browser and a simulator for a floating weather station. Annotation copyrighted by Book News, Inc., Portland, OR

A Quality of Service-driven Software Product Line Engineering Framework for Design and Analysis of Component-based Distributed Real-time and Embedded Systems

Model-Integrating Software Components

Software Product-line Engineering

Engineering Flexible Software Systems

From Use Cases to Pattern-based Software Architectures

First International Symposium, GCSE'99, Erfurt, Germany, September 28-30, 1999. Revised Papers

Software product lines are emerging as a critical new paradigm for software development. Product lines are enabling organizations to achieve impressive time-to-market gains and cost reductions. With the increasing number of product lines and product-line researchers and practitioners, the time is right for a comprehensive examination of the issues surrounding the software product line approach. The

Software Engineering Institute at Carnegie Mellon University is proud to sponsor the first conference on this important subject. This book comprises the proceedings of the First Software Product Line Conference (SPLC1), held August 28-31, 2000, in Denver, Colorado, USA. The twenty-seven papers of the conference technical program present research results and experience reports that cover all aspects of software product lines. Topics include business issues, enabling technologies, organizational issues, and life-cycle issues. Emphasis is placed on experiences in the development and fielding of product lines of complex systems, especially those that expose problems in the design, development, or evolution of software product lines. The book will be essential reading for researchers and practitioners alike.

Designing Software Product Lines with UML

Component-based Product Line Engineering with UML

Second International Conference, ECSA 2008 Paphos, Cyprus, September 29-October 1, 2008 Proceedings

Component-Based Software Engineering

Second International Conference, SPLC 2, San Diego, CA, USA, August 19-22, 2002. Proceedings